FLASH TYPE MEMORY, ITS CONTROLLING METHOD, STORAGE DEVICE, AND COMPUTER SYSTEM

Patent Number: JP11096779
Publication date: 1999-04-09
Inventor(s): TANAKA KAZUYA
Applicant(s): VICTOR CO OF JAPAN LTD
Requested Patent: JP11096779
Application Number: JP19970270520 19970917
Priority Number(s):
IPC Classification: G11C16/02; G06F12/02
EC Classification:
Equivalents:

Abstract

PROBLEM TO BE SOLVED: To reduce the number of times of rewriting a flash type memory and to enable executing a program on an expanded storage device.
SOLUTION: A header area and a program data area are arranged for each block in a physical address of a flash memory 12 side. However, a header area and a program data area for each block are arranged continuously in a logical address of a processor 10 side. When data included a program data area (n) is rewritten, both of a block of the data area (n) and a block of a corresponding head area are required to rewrite. However, the program data area (n) is in the same block with the corresponding head area when it is considered on the physical address. Therefore, only the block may be rewritten.

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平11－96779

(43)公開日　平成11年(1999)４月９日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ１１Ｃ　16/02		Ｇ１１Ｃ　17/00	６０１Ａ
Ｇ０６Ｆ　12/02	５１０	Ｇ０６Ｆ　12/02	５１０Ａ

審査請求　未請求　請求項の数11　ＦＤ　（全 18 頁）

(21)出願番号　特願平9－270520

(22)出願日　平成９年(1997)９月17日

(71)出願人　000004329
日本ビクター株式会社
神奈川県横浜市神奈川区守屋町３丁目12番地

(72)発明者　田中　和也
神奈川県横浜市神奈川区守屋町３丁目12番地　日本ビクター株式会社内

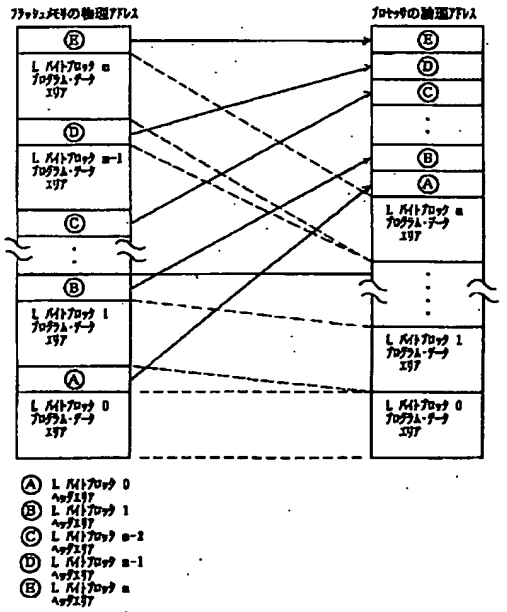
(74)代理人　弁理士　梶原　康稔

(54)【発明の名称】　フラッシュ型メモリ，その管理方法，記憶装置，コンピュータシステム

(57)【要約】

【課題】　フラッシュ型メモリの書換回数を低減するとともに、拡張記憶装置上でプログラムの実行を可能とする。

【解決手段】　フラッシュメモリ側の物理アドレスでは、各ブロック毎にヘッダエリア及びプログラム・データエリアを配置する。しかし、プロセッサ側の論理アドレス上では、各ブロックのヘッダエリア及びプログラム・データエリアを連続して配置する。プログラム・データエリアｎに含まれるデータを書き換えるときは、データエリアｎのブロックと、対応するヘッダエリアのブロックも書き換える必要がある。しかし、フラッシュメモリ１２の物理アドレス上でみると、プログラム・データエリアｎは該当するヘッダエリアとともに同一ブロック内にある。従って、そのブロックのみを書き換えればよい。



【特許請求の範囲】
【請求項１】　ブロック単位で記憶内容を消去するフラッシュ型メモリであって、
自己のブロックの管理情報を記憶する第１の領域と、主情報を記憶する第２の領域とを、各ブロック毎に設けたことを特徴とするフラッシュ型メモリ。
【請求項２】　請求項１記載のフラッシュ型メモリを備えた記憶装置であって、
隣接するブロックの第１の領域が、連続する論理アドレスとなるように物理アドレスと論理アドレスの変換を行うアドレス変換手段を備えたことを特徴とする記憶装置。
【請求項３】　請求項１記載のフラッシュ型メモリを備えた記憶装置であって、
隣接するブロックの第２の領域が、連続する論理アドレスとなるように物理アドレスと論理アドレスの変換を行うアドレス変換手段を備えたことを特徴とする請求項２記載の記憶装置。
【請求項４】　前記アドレス変換手段を、ワイヤードロジックによって構成したことを特徴とする請求項３記載の記憶装置。
【請求項５】　前記アドレス変換手段を、変換用のテーブルを格納したＲＯＭによって構成したことを特徴とする請求項３記載の記憶装置。
【請求項６】　前記管理情報は、そのブロックの書込回数と、そのブロックを消去した時刻を含むことを特徴とする請求項２又は３記載の記憶装置。
【請求項７】　ブロックをイレースする際に、書込回数が同じであれば、ブロックの消去時刻が古い方を優先的に選択する制御手段を備えたことを特徴とする請求項４記載の記憶装置。
【請求項８】　請求項３記載の記憶装置を備えたコンピュータシステムであって、
前記フラッシュ型メモリ上で、前記第２の領域に記憶されているプログラムを実行するプロセッサを含むことを特徴とするコンピュータシステム。
【請求項９】　ブロック単位で記憶内容を消去するフラッシュ型メモリの管理方法であって、
自己のブロックの管理情報を記憶する第１の領域と、主情報を記憶する第２の領域とを、各ブロック毎に設けることを特徴とするフラッシュ型メモリの管理方法。
【請求項１０】　隣接するブロックの第１の領域が連続する論理アドレスとなるように、物理アドレスと論理アドレスの変換を行うことを特徴とする請求項９記載のフラッシュ型メモリの管理方法。
【請求項１１】　隣接するブロックの第２の領域が連続する論理アドレスとなるように、物理アドレスと論理アドレスの変換を行うことを特徴とする請求項１０記載のフラッシュ型メモリの管理方法。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】この発明は、フラッシュ型メモリ，その管理方法，記憶装置，コンピュータシステムにかかり、更に具体的には、フラッシュ型メモリの効率的な管理手法の改良に関するものである。
【０００２】
【背景技術】一般にフラッシュタイプのフローティングゲートトランジスタを含む電気的に消去可能なプログラマブル読出専用メモリ（ＥＥＰＲＯＭ）は、現在市場で容易に入手できる。これらのいわゆるフラッシュメモリは、機能・性能面でＥＰＲＯＭメモリと類似した不揮発メモリであり、メモリ内で分割されているブロックを消去する回路内プログラマブル動作を可能にするという機能を更に有する。フラッシュメモリでは、以前に書き込まれたブロック領域を前もって消去することで、その内容の書き換えが行われる。
【０００３】典型的なコンピュータシステムでは、オペレーティングシステム（以下、単に「ＯＳ」という）のプログラムがそのシステムのデータ記憶装置のデータ管理を担う。ＯＳプログラムとの互換性を達成するために必要かつ十分であるデータ記憶装置のアトリビュート（属性）は、データ記憶装置のいかなる位置からもデータを読み出すことができ、これにデータを書き込むことができることである。しかし、フラッシュメモリの場合、データが既に書き込まれている領域には、その領域のデータを消去しなければデータを書き込むことはできない。このため、フラッシュメモリは、典型的な既存のＯＳプログラムとは互換性がない。
【０００４】このような点に着目し、既存のＯＳプログラムによってフラッシュメモリを管理することを可能にするソフトウエアが先行技術において提案されている。この先行技術のプログラムでは、フラッシュメモリを「書込み１回読出し複数回」の装置として動作させるか、「書込み複数回読出し複数回」の装置として動作させている。前者は、以前に書き込まれているメモリ領域を再利用することはできない装置であり、補助記憶装置や拡張記憶装置として使用できる。後者は、以前に書き込まれているメモリ領域を再利用可能とし、その領域中にはフラッシュメモリの書き換回数を少なくするような制御を持つ補助記憶装置（半導体ファイル記憶装置）がある。
【０００５】
【発明が解決しようとする課題】上述のような先行技術に見られる装置によれば、フラッシュメモリを使用した拡張記憶装置上では書換回数が少なくなるような制御構造を持たずにＲＯＭエグゼキュータブルなプログラムを動作させている。つまり、フラッシュメモリのデータを書き換えるときは、全ブロックを一括して消去し、その後、データを記憶させる必要がある。また、フラッシュメモリの書換回数を少なくするような制御構造を持つ装

置としては補助記憶装置（半導体ファイル記憶装置）があるものの、フラッシュメモリを使用した拡張記憶装置上でプログラムの実行を可能とする装置にはない。補助記憶装置において書換回数を少なくするためにその情報（書換回数テーブル）を異なるメモリ上に記憶する方式があるが、コスト高となる。また、同一フラッシュメモリ上に記憶する方式では、データを完全に連続的な配置できることができない。

【０００６】この発明は、以上の点に着目したもので、その目的は、フラッシュメモリを使用した拡張記憶装置上で、フラッシュメモリの書換回数を少なくするような制御構造を持つ装置を提供することである。また、他の目的は、主記憶装置にプログラムをロードすることなく、拡張記憶装置上でプログラムの実行を可能とすることである。更に他の目的は、フラッショメモリを使用した補助記憶装置として、書換回数を少なくする制御構造をもちながら、連続的な論理アドレスを不連続な物理アドレスに変換するメモリ管理装置によって、隣接するブロックのデータ領域ブロックを、アクセスのためのオーバヘッドを低減して、高速アクセス可能とすることである。

【０００７】

【課題を解決するための手段】前記目的を達成するため、本発明のフラッシュ型メモリは、自己のブロックの管理情報を記憶する第１の領域と、主情報を記憶する第２の領域とを、各ブロック毎に設けたことを特徴とする。本発明の記憶装置は、前記フラッシュ型メモリ、の隣接するブロックの第１の領域もしくは第２の領域が、連続する論理アドレスとなるように物理アドレスと論理アドレスの変換を行うアドレス変換手段を備えたことを特徴とする。前記アドレス変換手段は、例えば、ワイヤードロジックによって構成される。また、前記管理情報には、例えば、そのブロックの書込回数と、そのブロックを消去した時刻とが含まれる。他の発明は、ブロックをイレースする際に、書込回数が同じであれば、ブロックの消去時刻が古い方を優先的に選択する制御手段を備えたことを特徴とする。

【０００８】本発明のコンピュータシステムは、前記フラッシュ型メモリ上で、前記第２の領域に記憶されているプログラムを実行するプロセッサを含むことを特徴とする。本発明のフラッシュ型メモリの管理方法は、自己のブロックの管理情報を記憶する第１の領域と、主情報を記憶する第２の領域とを、各ブロック毎に設けることを特徴とする。主要な形態の一つは、隣接するブロックの第１の領域が連続する論理アドレスとなるように、物理アドレスと論理アドレスの変換を行うことを特徴とする。他の形態は、隣接するブロックの第２の領域が連続する論理アドレスとなるように、物理アドレスと論理アドレスの変換を行うことを特徴とする。

【０００９】この発明の前記及び他の目的，特徴，利点は、以下の詳細な説明及び添付図面から明瞭になろう。

【００１０】

【発明の実施の形態】以下、本発明の実施の形態について詳細に説明する。本発明は、例えば図１に示すように、プロセッサ１０，フラッシュメモリ１２およびその制御装置１４を含む拡張もしくは補助の記憶装置，ＲＡＭ（主記憶装置）１６を含むコンピュータシステムに適用される。フラッシュメモリ１２は、その書換回数を少なくなるようにするために、フラッシュメモリ１２のブロック単位の書換情報（イレース情報）を持つ領域（ヘッダエリア）と、フラッシュメモリ１２上のプログラムやデータを記憶する領域（プログラム・データエリア）を持つ（後述する図４参照）。フラッシュメモリ制御装置１４は、フラッシュメモリ１２へのリード，ライト，チップセレクトなどの信号を生成し、プロセッサ１０からフラッシュメモリ１２をインターフェース（Ｉ／Ｆ）する役割を担う。

【００１１】ヘッダエリアには、例えばイレースの回数や時刻を記述する。ここで、プロセッサ１０の論理アドレスをフラッシュメモリ１２のブロック数の単位に分割してフラッシュメモリ１２に記憶することができれば、フラッシュメモリ１２の書換回数の低減とフラッシュメモリ１２の管理方法が簡単になる。すなわち、書換えるべきデータが複数のブロックに跨っていると、それら複数のブロックについて書換処理を行わなければならない。しかし、書換えるべきデータが一つのブロック内にあれば、そのブロックのみを書換えればよく、フラッシュメモリ１２の書換回数を低減して管理を簡略化することができる。

【００１２】そこで、図２に示す装置を用いて、図３に示すメモリマッピングを行う。つまり、ヘッダエリアやプログラム・データエリアを、ブロック数の単位にそれぞれ分割する。そして分割された各エリアを、フラッシュメモリの中に効率よくブロック単位で配置する。すなわち、同時に書換えが必要となる分割ヘッダエリアと分割プログラム・データエリアが、同一のブロックに含まれるように配置する。このような配置は、プロセッサ１０の論理アドレスをフラッシュメモリ１２の物理アドレスに変換するアドレス変換装置２０によって実現する。

【００１３】図４に、ブロック構造を持つフラッシュメモリ１２のメモリマップの一例を示す。この例は、１ブロックがＬバイトで、かつ全体でｍ＋１ブロックをもつ、Ｌ×（ｍ＋１）バイトを記憶できるフラッシュメモリである。各ブロックを管理するためにフラッシュメモリ１２の書換回数などの情報を持つヘッダエリアとして、Ｐバイト必要であるとすると、全体でＰ×（ｍ＋１）バイトのヘッダエリアが必要である。従って、このシステムは、（Ｌ－Ｐ）×（ｍ＋１）バイトのプログラム・データエリアを持つことになる。

【００１４】このようなエリア構成のフラッシュメモリ

１２に対し、図２に示したアドレス変換装置１８による変換を行う。詳述すると、図５に示すように、プロセッサ１０の論理アドレスでは各ブロックのヘッダエリアを順にブロック０，ブロック１，……，ブロックｍのように連続して配置する。一方、プログラム・データエリアについては、順にブロック０，ブロック１，……，ブロックｍとヘッダエリア部分を抜いて連続した配置とする。これによって、各ブロックのヘッダエリアが、それぞれ連続する論理アドレスで配置されることになる。また同様に、各ブロックのプログラム・データエリアも、それぞれ連続する論理アドレスで配置されることになる。

【００１５】このようなアドレス配置において、例えばプログラム・データエリアｎに含まれるデータを書き換える必要が生じたとする。プロセッサ１０の論理アドレス上でみると、プログラム・データエリアｎのヘッダエリアが含まれるブロックと、プログラム・データエリアｎに相当するブロックの合計２つのブロックを書き換えなければならない。しかし、フラッシュメモリ１２の物理アドレス上でみると、プログラム・データエリアｎは該当するヘッダエリアとともに同一ブロック内にある。従って、そのブロックのみを書き換えればよい。

【００１６】このように、本形態によれば、フラッシュメモリの書換回数を低減する高速の管理プログラムがヘッダエリアの情報を読み、実行プログラムやデータの書換えや追記を簡便に実現できる。また、論理アドレス上でプログラム・データエリアのブロック間が連続的に配置しているため、無駄な空きスペースのないデータ構造を持つ拡張記憶装置および補助記憶装置を実現でき、かつフラッシュメモリ上での実行プログラムの動作が可能となる。更に、ヘッダエリアは、自己が管理するブロックの中に記憶されることになるため、ヘッダエリアのための別個のメモリ領域を必要としない。

【００１７】フラッシュメモリの各ブロックの管理方法を説明すると、最初はすべて、ヘッダエリアには「ＦＦｈ（１６進表示）」が記憶されている。フラッシュメモリは、初期化しなければ拡張記憶装置や補助記憶装置として使用できないため、ブロック０のヘッダエリアに書換回数０回と現時刻を記憶し、同様にブロック１のヘッダエリアに書換回数０回と現時刻を記憶し、……という具合に昇順で最後のブロックまで繰り返し記憶する。

【００１８】拡張記憶装置や補助記憶装置としてフラッシュメモリを使用するシステムでは、データや実行プログラムは追記させる構造でメモリに記憶させるのが通例である。この追記させる構造では、更新されたデータや実行プログラムは追記されるが、以前のデータや実行プログラムは無効となるような構造を持っている。そして、繰り返しデータや実行プログラムが更新されていくと、無駄なメモリエリアが拡大していく。そこで、フラッシュメモリ上のメモリ領域の再配置を行う必要がある。そのとき、イレースするブロックの候補として、書換回数の少ないブロック優先してイレースする。このとき、書換回数が同じブロックが複数あり、いずれかをイレースする必要があるときは、前回のブロック消去時刻の古い方を優先してイレースする。

【００１９】実際にブロックを消去するときであるが、そのブロックのヘッダエリアに記憶している以前の書換回数「Ｕ」を一時的に別メモリあるいはレジスタに保持し、その後フラッシュメモリのブロックを消去させるプログラムによって目的とするブロック（ヘッダエリア及びデータ・プログラムエリア）を消去して初期化する。その後すぐに、そのブロックのヘッダエリアに以前の書換回数に「１」インクリメントした値「Ｕ＋１」を書き込み、同時に現時刻も書き込む。そして必要に応じて、データ・プログラムを再配置する。以下、本形態の実施例について説明する。

【００２０】

【実施例１】図１に示したような、汎用ＣＰＵもしくは汎用マイコンであるプロセッサ１０，、高速にアクセス可能なＲＡＭ１６，フラッシュメモリ１２及びその制御装置１４を含む拡張記憶装置もしくは補助記憶装置をもつようなシステムにおいて、本実施例では、拡張記憶装置もしくは補助記憶装置に使用するフラッシュメモリ１２として、２メガビット×８で合計３２個の６４Ｋバイトのブロックをもつものを搭載したとする。

【００２１】このフラッシュメモリ１２は、図６に示すような物理アドレスのメモリマップを持つ。各ブロックは、ＯＳプログラム又は制御プログラムが管理するために、１ブロック毎に自己のヘッダエリアを持つ。ヘッダエリアには、その内容として、ブロックの消去回数や消去した時刻などを図７に示すような構造で記録されており、１６バイト分の容量をとる。なお、図７中に示すアドレスは、フラッシュメモリ１２の物理アドレスである。ブロックｎのヘッダエリアに記憶されている内容を参照して、ＯＳプログラム又は制御プログラムがブロックｎの消去や書換えを管理する。

【００２２】フラッシュメモリ１２の物理アドレスは、プロセッサ１０から見た論理アドレスに変換され、図３もしくは図５に示したように、各ブロックのヘッダエリアや、プログラム・データエリアのブロックが連続するようになる。この論理アドレスと物理アドレスとの変換の計算式の一例を示すと、以下のようになる。なお、数値ｎは、１０進数表記で０から３１までの数字である。このｎを１６進数に変換した数値をＮ（ｈ）とする。物理アドレスの値を「１６進数：［ＰＡ（ｈ）］，ＰＡ［２０：０］」とし、論理アドレスの値を「１６進数：［ＬＡ（ｈ）］，ＬＡ［２０：０］」とし、ブロックＮ（ｈ）を用いて変換式を示すと、以下のようになる。

【００２３】

```
IF (000000h≦LA (h) ≦00FFEFh)
        PA (h) =LA (h)
ELSE  IF (00FFF0h≦LA (h) ≦01FFDFh)
        PA (h) =LA (h) +10 (h)
ELSE  IF (00FFE0h≦LA (h) ≦02FFCFh)
        PA (h) =LA (h) +20 (h)
ELSE  IF (02FFD0h≦LA (h) ≦03FFBFh)
        PA (h) =LA (h) +30 (h)
ELSE  IF (03FFC0h≦LA (h) ≦04FFAFh)
        PA (h) =LA (h) +40 (h)
ELSE  IF (04FFB0h≦LA (h) ≦05FF9Fh)
        PA (h) =LA (h) +50 (h)
ELSE  IF (05FFA0h≦LA (h) ≦06FF8Fh)
        PA (h) =LA (h) +60 (h)
ELSE  IF (06FF90h≦LA (h) ≦07FF7Fh)
        PA (h) =LA (h) +70 (h)
ELSE  IF (07FF80h≦LA (h) ≦08FF6Fh)
        PA (h) =LA (h) +80 (h)
ELSE  IF (08FF70h≦LA (h) ≦09FF5Fh)
        PA (h) =LA (h) +90 (h)
ELSE  IF (09FF60h≦LA (h) ≦0AFF4Fh)
        PA (h) =LA (h) +A0 (h)
ELSE  IF (0AFF50h≦LA (h) ≦0BFF3Fh)
        PA (h) =LA (h) +B0 (h)
ELSE  IF (0BFF40h≦LA (h) ≦0CFF2Fh)
        PA (h) =LA (h) +C0 (h)
ELSE  IF (0CFF30h≦LA (h) ≦0DFF1Fh)
        PA (h) =LA (h) +D0 (h)
ELSE  IF (0DFF20h≦LA (h) ≦0EFF0Fh)
        PA (h) =LA (h) +E0 (h)
ELSE  IF (0EFF10h≦LA (h) ≦0FFEFFh)
        PA (h) =LA (h) +F0 (h)
ELSE  IF (0FFF00h≦LA (h) ≦10FEEFh)
        PA (h) =LA (h) +100 (h)
ELSE  IF (10FEF0h≦LA (h) ≦11FEDFh)
        PA (h) =LA (h) +110F (h)
ELSE  IF (11FEE0h≦LA (h) ≦12FECFh)
        PA (h) =LA (h) +120 (h)
ELSE  IF (12FED0h≦LA (h) ≦13FEBFh)
        PA (h) =LA (h) +130 (h)
ELSE  IF (13FEC0h≦LA (h) ≦14FEAFh)
        PA (h) =LA (h) +140 (h)
ELSE  IF (14FEB0h≦LA (h) ≦15FE9Fh)
        PA (h) =LA (h) +150 (h)
ELSE  IF (15FEA0h≦LA (h) ≦16FE8Fh)
        PA (h) =LA (h) +160 (h)
ELSE  IF (16FE90h≦LA (h) ≦17FE7Fh)
        PA (h) =LA (h) +170 (h)
ELSE  IF (17FE80h≦LA (h) ≦18FE6Fh)
        PA (h) =LA (h) +180 (h)
```

```
ELSE  IF (18FE70h≦LA (h) ≦19FE5Fh)
          PA (h) =LA (h) +190 (h)
ELSE  IF (19FE60h≦LA (h) ≦1AFE4Fh)
          PA (h) =LA (h) +1A0 (h)
ELSE  IF (1AFE50h≦LA (h) ≦1BFE3Fh)
          PA (h) =LA (h) +1B0 (h)
ELSE  IF (1BFE40h≦LA (h) ≦1CFE2Fh)
          PA (h) =LA (h) +1C0 (h)
ELSE  IF (1CFE30h≦LA (h) ≦1DFE1Fh)
          PA (h) =LA (h) +1D0 (h)
ELSE  IF (1DFE20h≦LA (h) ≦1EFE0Fh)
          PA (h) =LA (h) +1E0 (h)
ELSE  IF (1EFE10h≦LA (h) ≦1FFDFFh)
          PA (h) =LA (h) +1F0 (h)
ELSE  IF (1FFE00h≦LA (h) ≦1FFE0Fh)
          PA (h) =LA (h) -1EFE10 (h)
ELSE  IF (1FFE10h≦LA (h) ≦1FFE1Fh)
          PA (h) =LA (h) -1DFE20 (h)
ELSE  IF (1FFE20h≦LA (h) ≦1FFE2Fh)
          PA (h) =LA (h) -1CFE30 (h)
ELSE  IF (1FFE30h≦LA (h) ≦1FFE3Fh)
          PA (h) =LA (h) -1BFE40 (h)
ELSE  IF (1FFE40h≦LA (h) ≦1FFE4Fh)
          PA (h) =LA (h) -1AFE50 (h)
ELSE  IF (1FFE50h≦LA (h) ≦1FFE5Fh)
          PA (h) =LA (h) -19FE60 (h)
ELSE  IF (1FFE60h≦LA (h) ≦1FFE6Fh)
          PA (h) =LA (h) -18FE70 (h)
ELSE  IF (1FFE70h≦LA (h) ≦1FFE7Fh)
          PA (h) =LA (h) -17FE80 (h)
ELSE  IF (1FFE80h≦LA (h) ≦1FFE8Fh)
          PA (h) =LA (h) -16FE90 (h)
ELSE  IF (1FFE90h≦LA (h) ≦1FFE9Fh)
          PA (h) =LA (h) -15FEA0 (h)
ELSE  IF (1FFEA0h≦LA (h) ≦1FFEAFh)
          PA (h) =LA (h) -14FEB0 (h)
ELSE  IF (1FFEB0h≦LA (h) ≦1FFEBFh)
          PA (h) =LA (h) -13FEC0 (h)
ELSE  IF (1FFEC0h≦LA (h) ≦1FFECFh)
          PA (h) =LA (h) -12FED0 (h)
ELSE  IF (1FFED0h≦LA (h) ≦1FFEDFh)
          PA (h) =LA (h) -11FEE0 (h)
ELSE  IF (1FFEE0h≦LA (h) ≦1FFEEFh)
          PA (h) =LA (h) -10FEF0 (h)
ELSE  IF (1FFEF0h≦LA (h) ≦1FFEFFh)
          PA (h) =LA (h) -FFF00 (h)
ELSE  IF (1FFF00h≦LA (h) ≦1FFF0Fh)
          PA (h) =LA (h) -EFF10 (h)
ELSE  IF (1FFF10h≦LA (h) ≦1FFF1Fh)
          PA (h) =LA (h) -DFF20 (h)
```

```
ELSE  IF (1FFF20h≦LA (h) ≦1FFF2Fh)
          PA (h) =LA (h) −CFF30 (h)
ELSE  IF (1FFF30h≦LA (h) ≦1FFF3Fh)
          PA (h) =LA (h) −BFF40 (h)
ELSE  IF (1FFF40h≦LA (h) ≦1FFF4Fh)
          PA (h) =LA (h) −AFF50 (h)
ELSE  IF (1FFF50h≦LA (h) ≦1FFF5Fh)
          PA (h) =LA (h) −9FF60 (h)
ELSE  IF (1FFF60h≦LA (h) ≦1FFF6Fh)
          PA (h) =LA (h) −8FF70 (h)
ELSE  IF (1FFF70h≦LA (h) ≦1FFF7Fh)
          PA (h) =LA (h) −7FF80 (h)
ELSE  IF (1FFF80h≦LA (h) ≦1FFF8Fh)
          PA (h) =LA (h) −6FF90 (h)
ELSE  IF (1FFF90h≦LA (h) ≦1FFF9Fh)
          PA (h) =LA (h) −5FFA0 (h)
ELSE  IF (1FFFA0h≦LA (h) ≦1FFFAFh)
          PA (h) =LA (h) −4FFB0 (h)
ELSE  IF (1FFFB0h≦LA (h) ≦1FFFBFh)
          PA (h) =LA (h) −3FFC0 (h)
ELSE  IF (1FFFC0h≦LA (h) ≦1FFFCFh)
          PA (h) =LA (h) −2FFD0 (h)
ELSE  IF (1FFFD0h≦LA (h) ≦1FFFDFh)
          PA (h) =LA (h) −1FFE0 (h)
ELSE  IF (1FFFE0h≦LA (h) ≦1FFFEFh)
          PA (h) =LA (h) −FFF0 (h)
ELSE  IF (1FFFF0h≦LA (h) ≦1FFFFFh)
          PA (h) =LA (h)
```

【００２４】一例を示すと、最初の

```
IF (000000h≦LA (h) ≦00FFEFh)
          PA (h) =LA (h)
```

は、論理アドレスＬＡ（ｈ）が０００００ｈと００ＦＦＥＦｈの間にあるときに、物理アドレスＰＡ（ｈ）は論理アドレスＬＡ（ｈ）と同じ値であることを意味する。

【００２５】同様に、

```
ELSE  IF (00FFF0h≦LA (h) ≦01FFDFh)
          PA (h) =LA (h) +10 (h)
```

は、論理アドレスＬＡ（ｈ）が００ＦＦＦ０ｈと０１ＦＦＤＦｈの間にあるときに、物理アドレスＰＡ（ｈ）は論理アドレスＬＡ（ｈ）に１０（ｈ）加算した値であることを意味する。以下、同様である。

【００２６】次に、図２のアドレス変換装置１８を、上述したアドレス変換式を用いてワイヤードロジックで構成し、フラッシュメモリ１２の制御装置１４とともにプロセッサ１０とＩ／Ｆをとる。本実施例のフラッシュメモリ１２のブロック管理方法を説明すると、上述したように、最初に各ブロックのヘッダエリアの初期化（イニシャライズ）を行う。初期化の一例を示すと、次のようになる。

【００２７】ブロック０のヘッダエリアでは、物理アドレスＰＡ［２０：０］の「００ＦＦＦＤｈ」から「００ＦＦＦＦｈ」にそれぞれ「００ｈ」を書き込み、「００ＦＦＦ６ｈ」から「００ＦＦＦＢｈに」現在の時刻をそれぞれ書き込む。次に、ブロック１のヘッダエリアでは、物理アドレスＰＡ［２０：０］の「０１ＦＦＦＤｈ」から「０１ＦＦＦＦｈ」にそれぞれ「００ｈ」を書き込み、「０１ＦＦＦ６ｈ」から「０１ＦＦＦＢｈ」に現在の時刻をそれぞれ書き込む。……そして、昇順ブロックｘｘｈのヘッダエリアでは、物理アドレスＰＡ［２０：０］の「ｘｘＦＦＦＤｈ」から「ｘｘＦＦＦＦｈ」にそれぞれ「００ｈ」を書き込み、「ｘｘＦＦＦ６ｈ」から「ｘｘＦＦＦＢｈ」に現在の時刻をそれぞれ書き込む。……最終ブロック１Ｆｈのヘッダエリアでは、物理アドレスＰＡ［２０：０］の「１ＦＦＦＦＤｈ」から「１ＦＦＦＦＦｈ」にそれぞれ「００ｈ」を書き込み、「１ＦＦＦＦ６ｈ」から「１ＦＦＦＦＢｈ」に現在の時刻をそれぞれ書き込んで初期化を終了する。

【００２８】拡張記憶装置あるいは補助記憶装置としてフラッシュメモリを用いた本システムでは、データや実行プログラムなどは追記する手法で新たに更新していくことになる。この手法自体は従来の手法と同様である。しかし、データや実行プログラムの更新を繰り返す度に、不要なメモリエリアが拡大していく。そこで、データや実行プログラムの更新により不要となったデータ・実行プログラムやメモリフラグメンテーションを削除するために、メモリデータの再配置（デフラグメンテーション）を行い、不要なメモリエリアの縮小と再利用エリアの拡大を図るようにする。このとき、イレースするブロックの候補として、上述したように書換回数の少ない方のブロックを優先して消去する。もし、書換回数が同じで２つ以上の候補があるときは、その候補の以前の消去時刻の古いブロックを優先してイレースすることにする。この処理は、デフラグメンテーションプログラムにより、プロセッサ１０がフラッシュメモリ制御装置１４を介して行う。

【００２９】具体的には、優先的に選ばれたブロックのヘッダエリアに記憶されている書換回数「Ｖ」を他のメモリあるいはレジスタに記憶し、そして次に該当ブロックをイレースするためのプログラム（イレースシーケンスルーチン）の実行に移る。このプログラムが終了後、そのブロックはイレースされたことになる。そして直ちに、ヘッダエリアの更新を行う。すなわち、書換回数を以前の回数に１インクリメントした値「Ｖ＋１」として新たに書き込み、更に現在の時刻を書き込む。この作業の終了後、通常のデータなどの追記や読み出しができる状態に戻す。

【００３０】

【実施例２】この実施例２では、拡張記憶装置もしくは補助記憶装置に使用するフラッシュメモリ１２として、１メガビット×１６で合計３２個の３２キロワードのブロックをもつものを搭載したシステム構成となっている。このフラッシュメモリは、図８に示すような物理アドレスのメモリマップを持つ。前記実施例と同様に、各ブロックは、ＯＳプログラム又は制御プログラムが管理するために、１ブロック毎に自己のヘッダエリアを持つ。その内容であるブロックの消去回数や消去した時刻などは、図９に示すような構造で８ワード分をとる。なお、図９中に示すアドレスは、フラッシュメモリ１２の物理アドレスである。ブロックｎのヘッダエリアに記憶されている内容を参照して、ＯＳプログラム又は制御プログラムがブロックｎのブロック消去や書き換えを管理する。この管理方法は、前記実施例１と同様の手法で対応できる。

【００３１】次に、各ブロックのヘッダエリアやプログラム・データエリアのブロックが連続するように、アドレス変換を行う。この論理アドレスから物理アドレスへのアドレス変換の計算式の一例を示すと、以下のようになる。なお、数値ｎ，数値Ｎ（ｈ），物理アドレスの値［ＰＡ（ｈ）］，ＰＡ［２０：１］，論理アドレスの値［ＬＡ（ｈ）］，ＬＡ［２０：１］は、前記実施例と同様である。

【００３２】

```
IF (00000h≦LA (h) ≦07FF7h)
        PA (h) =LA (h)
ELSE  IF (07FF8h≦LA (h) ≦0FFEFh)
        PA (h) =LA (h) +8 (h)
ELSE  IF (0FFF0h≦LA (h) ≦17FE7h)
        PA (h) =LA (h) +10 (h)
ELSE  IF (17FE8h≦LA (h) ≦1FFDFh)
        PA (h) =LA (h) +18 (h)
ELSE  IF (1FFE0h≦LA (h) ≦27FD7h)
        PA (h) =LA (h) +20 (h)
ELSE  IF (27FD8h≦LA (h) ≦2FFCFh)
        PA (h) =LA (h) +28 (h)
ELSE  IF (2FFD0h≦LA (h) ≦37FC7h)
        PA (h) =LA (h) +30 (h)
ELSE  IF (37FC8h≦LA (h) ≦3FFBFh)
        PA (h) =LA (h) +38 (h)
ELSE  IF (3FFC0h≦LA (h) ≦47FB7h)
        PA (h) =LA (h) +40 (h)
ELSE  IF (47FB8h≦LA (h) ≦4FFAFh)
        PA (h) =LA (h) +48 (h)
ELSE  IF (4FFB0h≦LA (h) ≦57FA7h)
        PA (h) =LA (h) +50 (h)
```

```
ELSE  IF (57FA8h≦LA (h) ≦5FF9Fh)
          PA (h) =LA (h) +58 (h)
ELSE  IF (5FFA0h≦LA (h) ≦67F97h)
          PA (h) =LA (h) +60 (h)
ELSE  IF (67F98h≦LA (h) ≦6FF8Fh)
          PA (h) =LA (h) +68 (h)
ELSE  IF (6FF90h≦LA (h) ≦77F87h)
          PA (h) =LA (h) +70 (h)
ELSE  IF (77F88h≦LA (h) ≦7FF7Fh)
          PA (h) =LA (h) +78 (h)
ELSE  IF (7FF80h≦LA (h) ≦87F77h)
          PA (h) =LA (h) +80 (h)
ELSE  IF (87F78h≦LA (h) ≦8FF6Fh)
          PA (h) =LA (h) +88 (h)
ELSE  IF (8FF70h≦LA (h) ≦97F67h)
          PA (h) =LA (h) +90 (h)
ELSE  IF (97F68h≦LA (h) ≦9FF5Fh)
          PA (h) =LA (h) +98 (h)
ELSE  IF (9FF60h≦LA (h) ≦A7F57h)
          PA (h) =LA (h) +A0 (h)
ELSE  IF (A7F58h≦LA (h) ≦AFF4Fh)
          PA (h) =LA (h) +A8 (h)
ELSE  IF (AFF50h≦LA (h) ≦B7F47h)
          PA (h) =LA (h) +B0 (h)
ELSE  IF (B7F48h≦LA (h) ≦BFF3Fh)
          PA (h) =LA (h) +B8 (h)
ELSE  IF (BFF40h≦LA (h) ≦C7F37h)
          PA (h) =LA (h) +C0 (h)
ELSE  IF (C7F38h≦LA (h) ≦CFF2Fh)
          PA (h) =LA (h) +C8 (h)
ELSE  IF (CFF30h≦LA (h) ≦D7F27h)
          PA (h) =LA (h) +D0 (h)
ELSE  IF (D7F28h≦LA (h) ≦DFF1Fh)
          PA (h) =LA (h) +D8 (h)
ELSE  IF (DFF20h≦LA (h) ≦E7F17h)
          PA (h) =LA (h) +E0 (h)
ELSE  IF (E7F18h≦LA (h) ≦EFF0Fh)
          PA (h) =LA (h) +E8 (h)
ELSE  IF (EFF10h≦LA (h) ≦F7F07h)
          PA (h) =LA (h) +F0 (h)
ELSE  IF (F7F08h≦LA (h) ≦FFEFFh)
          PA (h) =LA (h) +F8 (h)
ELSE  IF (FFF00h≦LA (h) ≦FFF07h)
          PA (h) =LA (h) -F7F08h
ELSE  IF (FFF08h≦LA (h) ≦FFF0Fh)
          PA (h) =LA (h) -EFF10h
ELSE  IF (FFF10h≦LA (h) ≦FFF17h)
          PA (h) =LA (h) -E7F18h
ELSE  IF (FFF18h≦LA (h) ≦FFF1Fh)
          PA (h) =LA (h) -DFF20h
```

```
ELSE IF (FFF20h≦LA (h) ≦FFF27h)
        PA (h) =LA (h) −D7F28h
ELSE IF (FFF28h≦LA (h) ≦FFF2Fh)
        PA (h) =LA (h) −CFF30h
ELSE IF (FFF30h≦LA (h) ≦FFF37h)
        PA (h) =LA (h) −C7F38h
ELSE IF (FFF38h≦LA (h) ≦FFF3Fh)
        PA (h) =LA (h) −BFF40h
ELSE IF (FFF40h≦LA (h) ≦FFF47h)
        PA (h) =LA (h) −B7F48h
ELSE IF (FFF48h≦LA (h) ≦FFF4Fh)
        PA (h) =LA (h) −AFF50h
ELSE IF (FFF50h≦LA (h) ≦FFF57h)
        PA (h) =LA (h) −A7F58h
ELSE IF (FFF58h≦LA (h) ≦FFF5Fh)
        PA (h) =LA (h) −9FF60h
ELSE IF (FFF60h≦LA (h) ≦FFF67h)
        PA (h) =LA (h) −97F68h
ELSE IF (FFF68h≦LA (h) ≦FFF6Fh)
        PA (h) =LA (h) −8FF70h
ELSE IF (FFF70h≦LA (h) ≦FFF77h)
        PA (h) =LA (h) −87F78h
ELSE IF (FFF78h≦LA (h) ≦FFF7Fh)
        PA (h) =LA (h) −7FF80h
ELSE IF (FFF80h≦LA (h) ≦FFF87h)
        PA (h) =LA (h) −77F88h
ELSE IF (FFF88h≦LA (h) ≦FFF8Fh)
        PA (h) =LA (h) −6FF90h
ELSE IF (FFF90h≦LA (h) ≦FFF97h)
        PA (h) =LA (h) −67F98h
ELSE IF (FFF98h≦LA (h) ≦FFF9Fh)
        PA (h) =LA (h) −5FFA0h
ELSE IF (FFFA0h≦LA (h) ≦FFFA7h)
        PA (h) =LA (h) −57FA8h
ELSE IF (FFFA8h≦LA (h) ≦FFFAFh)
        PA (h) =LA (h) −4FFB0h
ELSE IF (FFFB0h≦LA (h) ≦FFFB7h)
        PA (h) =LA (h) −47FB8h
ELSE IF (FFFB8h≦LA (h) ≦FFFBFh)
        PA (h) =LA (h) −3FFC0h
ELSE IF (FFFC0h≦LA (h) ≦FFFC7h)
        PA (h) =LA (h) −37FC8h
ELSE IF (FFFC8h≦LA (h) ≦FFFCFh)
        PA (h) =LA (h) −2FFD0h
ELSE IF (FFFD0h≦LA (h) ≦FFFD7h)
        PA (h) =LA (h) −27FD8h
ELSE IF (FFFD8h≦LA (h) ≦FFFDFh)
        PA (h) =LA (h) −1FFE0h
ELSE IF (FFFE0h≦LA (h) ≦FFFE7h)
        PA (h) =LA (h) −17FE8h
```

ELSE IF (FFFE8h≦LA (h) ≦FFFEFh)
PA (h) =LA (h) −FFF0h
ELSE IF (FFFF0h≦LA (h) ≦FFFF7h)
PA (h) =LA (h) −7FF8h
ELSE IF (FFFF8h≦LA (h) ≦FFFFFh)
PA (h) =LA (h)

【0033】次に、図2のアドレス変換装置18を、上述したアドレス変換式を用いてワイヤードロジックで構成し、フラッシュメモリ12の制御装置14とともにプロセッサ10とI／Fをとる。このようにして、前記実施例と同様に、ヘッダエリアにおける書換えの管理・制御とプログラム・データエリアのブロック間を連続化するアドレスマップへの変換が可能となる。

【0034】

【他の実施例】この発明には数多くの実施の形態があり、以上の開示に基づいて多様に改変することが可能である。例えば、次のようなものも含まれる。

(1)前記実施例1，2では、ハードロジックによってアドレス変換装置を実現したが、アドレス変換用のROMを使用し、テーブル参照方式によって論理アドレスから物理アドレスへのアドレス変換を行うようにしてもよい。すなわち、論理アドレスをROMのアドレス側に供給し、物理アドレスをROMのデータ側から出力する。ROMのチップイネーブルやアウトプットイネーブルはアサートしたままにして、アドレスコントロールとする。このROMに、前記実施例で示したようなアドレス変換式を展開したテーブルを記憶すればよい。

【0035】(2)前記形態はフラッシュメモリに本発明を適用したものであるが、他の類似するメモリがフラッシュメモリと同じ書込み，読出し機能を備えており、かつ、書込前ブロック消去特性を有するメモリであれば、同様に適用可能である。

(3)前記実施例は、バイト単位やワード（16ビットや32ビット）単位による操作の場合であるが、他のデータ単位であっても、同様に適用可能である。

【0036】

【発明の効果】以上説明したように、本発明によれば、次のような効果がある。

(1)対応するヘッダエリアとプログラム・データエリアが、メモリの同一ブロック内となるようにアドレス変換を行うので、ブロックの書換回数を低減して、高速のアクセスが可能となる。また、ヘッダエリア用のメモリを必要とせず、コストの削減を可能にする。

(2)アドレス変換によって、プロセッサ側からみたときにプログラム・データエリアのブロックが連続するようになり、メモリ領域の無駄を省くとともに、主記憶装置に実行プログラムをロードすることなく、拡張記憶装置もしくは補助記憶装置上でプログラムの実行が可能となる。

【図面の簡単な説明】

【図1】本発明を利用したシステムの一例のブロック図である。

【図2】本発明の一形態の主要部を示すブロック図である。

【図3】論理アドレスから物理アドレスにアドレス変換するための、メモリマッピングの手法を示す概念図である。

【図4】ブロック消去機能をもつ汎用のフラッシュメモリにおける物理アドレスのメモリマッピングと、ヘッダエリア及びプログラム・データエリアの関係を示す図である。

【図5】フラッシュメモリの物理アドレスとプロセッサの論理アドレスの関係を示す図である。

【図6】実施例1におけるフラッシュメモリのメモリマップを示す図である。

【図7】実施例1におけるフラッシュメモリのブロックnにおける詳細な構造のメモリマップを示す図である。

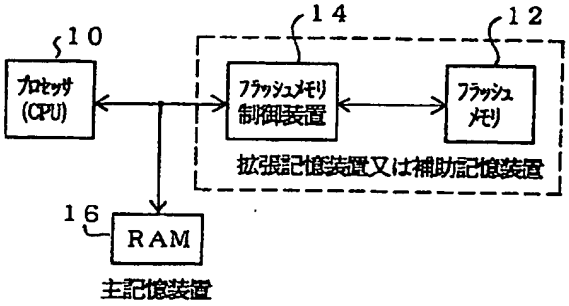
【図8】実施例2におけるフラッシュメモリのメモリマップを示す図である。

【図9】実施例2におけるフラッシュメモリのブロックnにおける詳細な構造のメモリマップを示す図である。

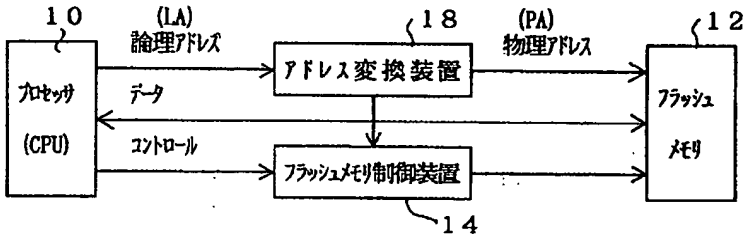
【符号の説明】

10…プロセッサ
12…フラッシュメモリ
14…フラッシュメモリ制御装置
16…RAM
18…アドレス変換装置

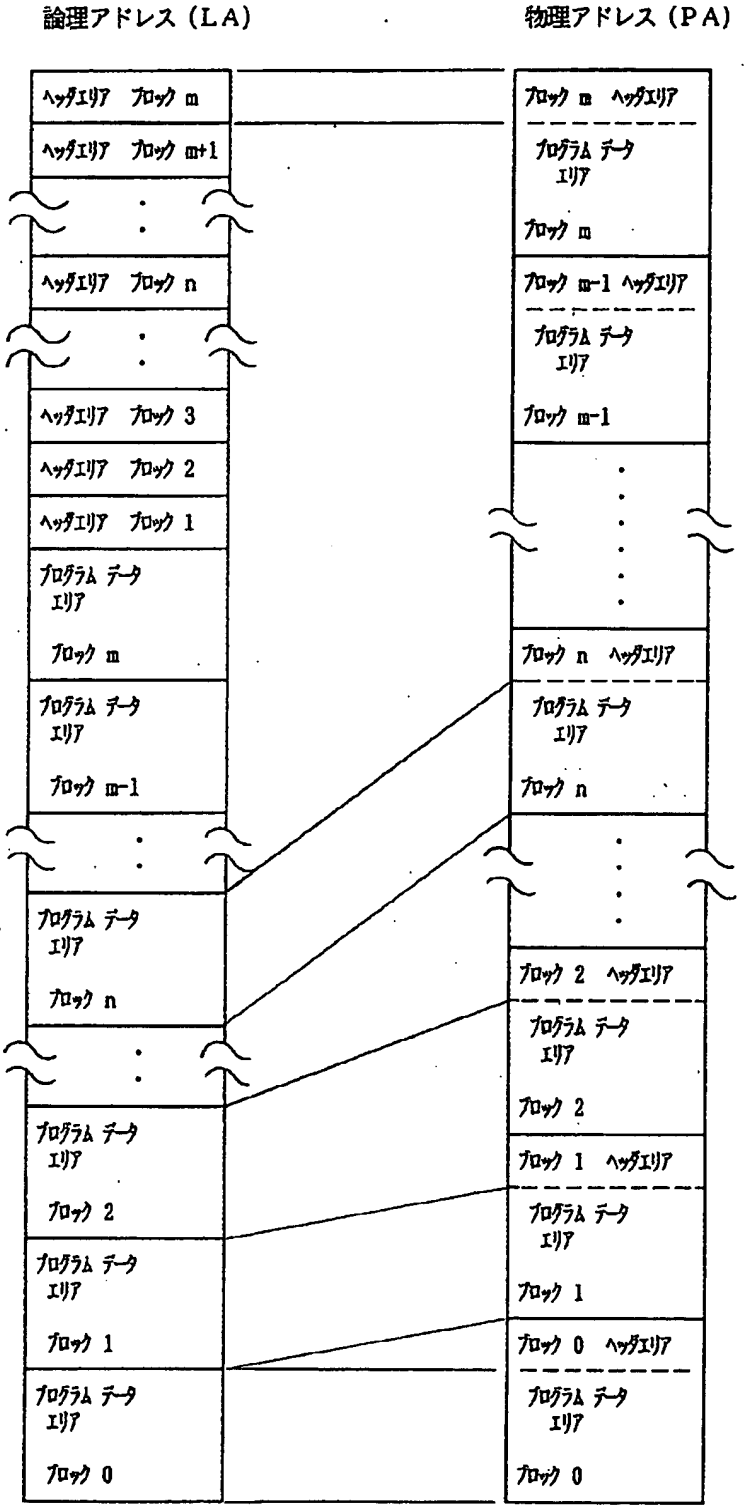
【図１】


１０
プロセッサ
(CPU)
１４
フラッシュメモリ
制御装置
１２
フラッシュ
メモリ
拡張記憶装置又は補助記憶装置
１６
RAM
主記憶装置

【図２】


１０
プロセッサ
(CPU)
(LA)
論理アドレス
データ
コントロール
１８
アドレス変換装置
フラッシュメモリ制御装置
１４
(PA)
物理アドレス
１２
フラッシュ
メモリ

【図３】

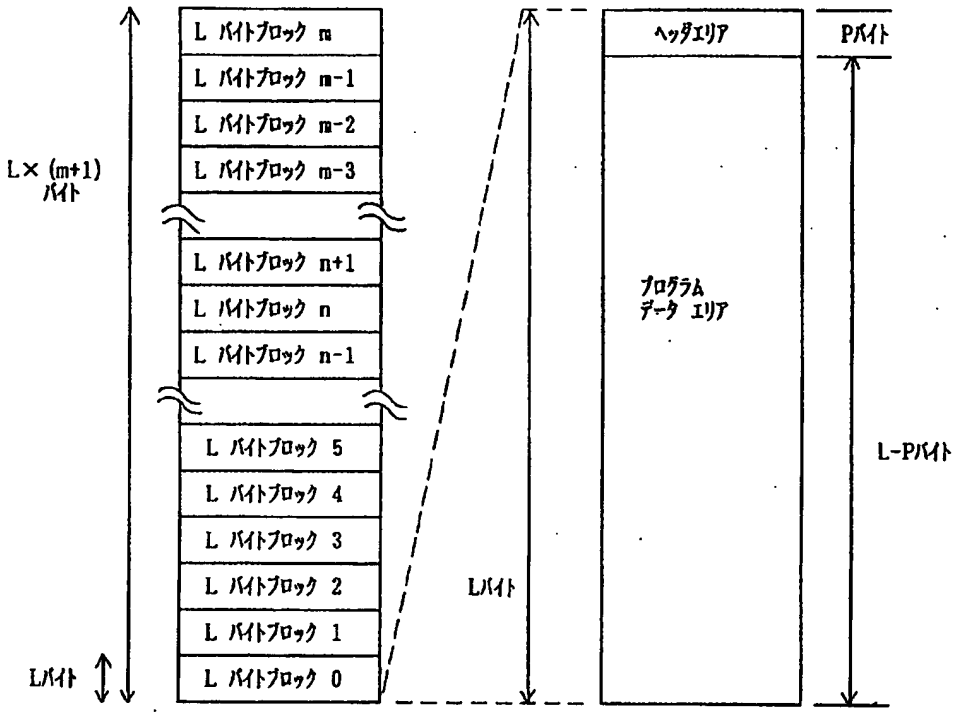


【図８】

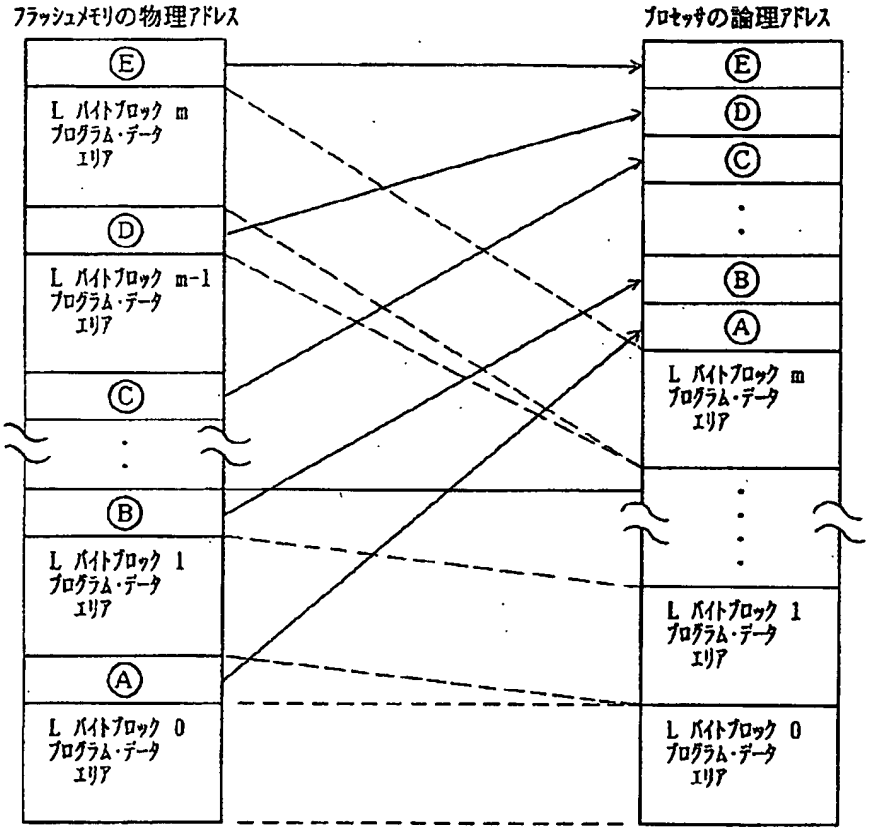
PA[20:0]		
FFFFF	F0000	32K ワード・ブロック 31
F7FFF	F0000	32K ワード・ブロック 30
EFFFF	E8000	32K ワード・ブロック 29
E7FFF	E0000	32K ワード・ブロック 28
DFFFF	D8000	32K ワード・ブロック 27
D7FFF	D0000	32K ワード・ブロック 26
CFFFF	C8000	32K ワード・ブロック 25
C7FFF	C0000	32K ワード・ブロック 24
BFFFF	B8000	32K ワード・ブロック 23
B7FFF	B0000	32K ワード・ブロック 22
AFFFF	A8000	32K ワード・ブロック 21
A7FFF	A0000	32K ワード・ブロック 20
9FFFF	98000	32K ワード・ブロック 19
97FFF	90000	32K ワード・ブロック 18
8FFFF	88000	32K ワード・ブロック 17
87FFF	80000	32K ワード・ブロック 16
7FFFF	78000	32K ワード・ブロック 15
77FFF	70000	32K ワード・ブロック 14
6FFFF	68000	32K ワード・ブロック 13
67FFF	60000	32K ワード・ブロック 12
5FFFF	58000	32K ワード・ブロック 11
57FFF	50000	32K ワード・ブロック 10
4FFFF	48000	32K ワード・ブロック 9
47FFF	40000	32K ワード・ブロック 8
3FFFF	38000	32K ワード・ブロック 7
37FFF	30000	32K ワード・ブロック 6
2FFFF	28000	32K ワード・ブロック 5
27FFF	20000	32K ワード・ブロック 4
1FFFF	18000	32K ワード・ブロック 3
17FFF	10000	32K ワード・ブロック 2
0FFFF	08000	32K ワード・ブロック 1
07FFF	00000	32K ワード・ブロック 0

ワード幅モード（16ビット）

【図４】


L バイトブロック m
L バイトブロック m-1
L バイトブロック m-2
L バイトブロック m-3
L バイトブロック n+1
L バイトブロック n
L バイトブロック n-1
L バイトブロック 5
L バイトブロック 4
L バイトブロック 3
L バイトブロック 2
L バイトブロック 1
L バイトブロック 0
L×(m+1) バイト
Lバイト
Lバイト
ヘッダエリア
Pバイト
プログラム データ エリア
L-Pバイト

【図５】



Ⓐ L バイトブロック 0 ヘッダエリア
Ⓑ L バイトブロック 1 ヘッダエリア
Ⓒ L バイトブロック m-2 ヘッダエリア
Ⓓ L バイトブロック m-1 ヘッダエリア
Ⓔ L バイトブロック m ヘッダエリア

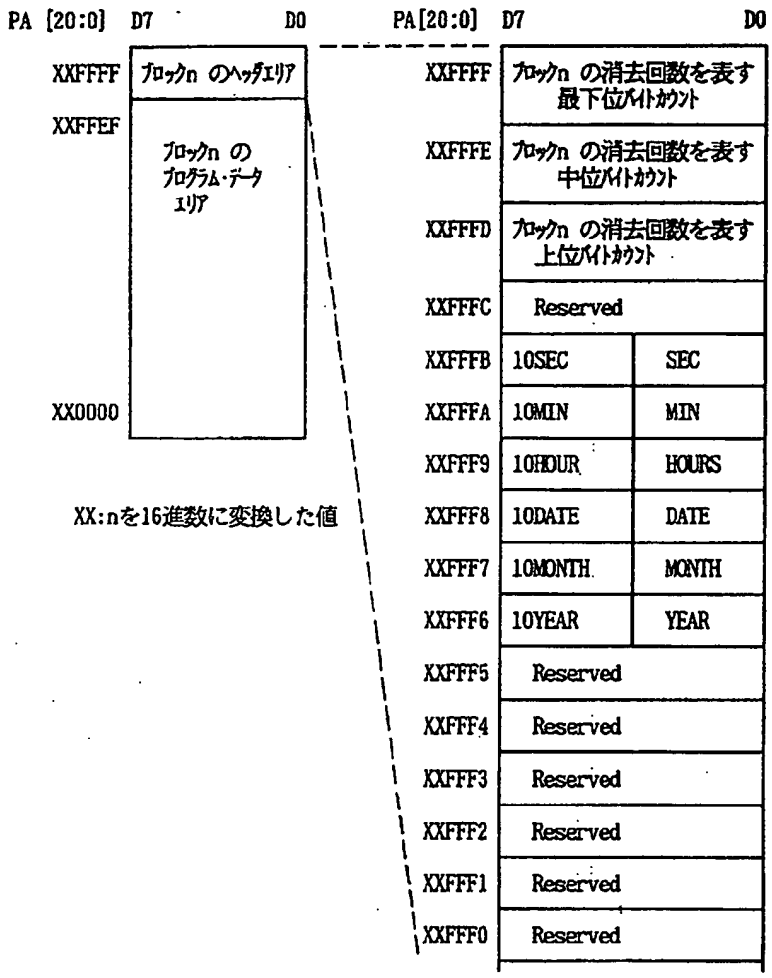
【図６】

PA[20:0]

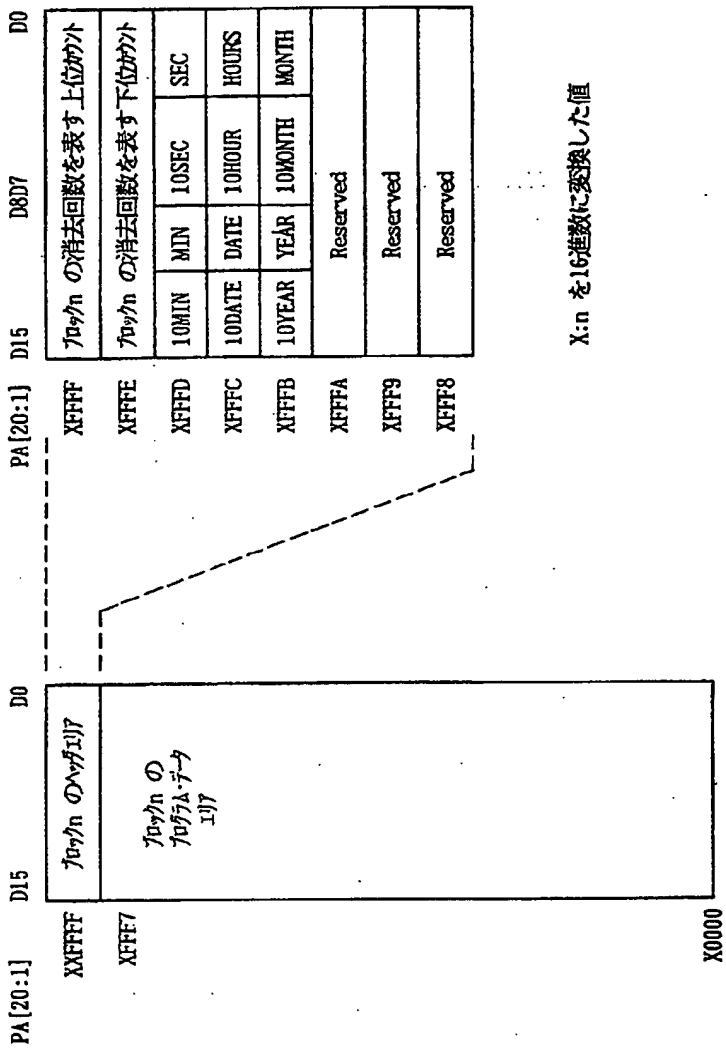
1FFFFF 1F0000	64K バイト・ブロック	31
1EFFFF 1E0000	64K バイト・ブロック	30
1DFFFF 1D0000	64K バイト・ブロック	29
1CFFFF 1C0000	64K バイト・ブロック	28
1BFFFF 1B0000	64K バイト・ブロック	27
1AFFFF 1A0000	64K バイト・ブロック	26
19FFFF 190000	64K バイト・ブロック	25
18FFFF 180000	64K バイト・ブロック	24
17FFFF 170000	64K バイト・ブロック	23
16FFFF 160000	64K バイト・ブロック	22
15FFFF 150000	64K バイト・ブロック	21
14FFFF 140000	64K バイト・ブロック	20
13FFFF 130000	64K バイト・ブロック	19
12FFFF 120000	64K バイト・ブロック	18
11FFFF 110000	64K バイト・ブロック	17
10FFFF 100000	64K バイト・ブロック	16
0FFFFF 0F0000	64K バイト・ブロック	15
0EFFFF 0E0000	64K バイト・ブロック	14
0DFFFF 0D0000	64K バイト・ブロック	13
0CFFFF 0C0000	64K バイト・ブロック	12
0BFFFF 0B0000	64K バイト・ブロック	11
0AFFFF 0A0000	64K バイト・ブロック	10
09FFFF 090000	64K バイト・ブロック	9
08FFFF 080000	64K バイト・ブロック	8
07FFFF 070000	64K バイト・ブロック	7
06FFFF 060000	64K バイト・ブロック	6
05FFFF 050000	64K バイト・ブロック	5
04FFFF 040000	64K バイト・ブロック	4
03FFFF 030000	64K バイト・ブロック	3
02FFFF 020000	64K バイト・ブロック	2
01FFFF 010000	64K バイト・ブロック	1
00FFFF 000000	64K バイト・ブロック	0

バイト幅モード(8ビット)

【図７】


PA [20:0]
D7
D0
XXFFFF
ブロックn のヘッダエリア
XXFFEF
ブロックn の
プログラム・データ
エリア
XX0000
XX:nを16進数に変換した値
PA[20:0]
D7
D0
XXFFFF
ブロックn の消去回数を表す
最下位バイトカウント
XXFFFE
ブロックn の消去回数を表す
中位バイトカウント
XXFFFD
ブロックn の消去回数を表す
上位バイトカウント
XXFFFC
Reserved
XXFFFB
10SEC
SEC
XXFFFA
10MIN
MIN
XXFFF9
10HOUR
HOURS
XXFFF8
10DATE
DATE
XXFFF7
10MONTH
MONTH
XXFFF6
10YEAR
YEAR
XXFFF5
Reserved
XXFFF4
Reserved
XXFFF3
Reserved
XXFFF2
Reserved
XXFFF1
Reserved
XXFFF0
Reserved

【図9】


PA[20:1]
D15
D0
XXFFFF
XFFF7
X0000
ブロックn のヘッダエリア
ブロックn のプログラム・データエリア
PA[20:1]
D15
D8D7
D0
XFFFF
XFFFE
XFFFD
XFFFC
XFFFB
XFFFA
XFFF9
XFFF8
ブロックn の消去回数を表す上位カウント
ブロックn の消去回数を表す下位カウント
10MIN
MIN
10SEC
SEC
10DATE
DATE
10HOUR
HOURS
10YEAR
YEAR
10MONTH
MONTH
Reserved
Reserved
Reserved
X:n を16進数に変換した値